AMENDMENTS TO THE CLAIMS

Please amend the claims as follows.

1. (Currently Amended) A method of processing a network connection in a computer system, comprising:

   establishing the network connection by a [[first]] software network protocol stack implemented in the kernel of an operating system associated with the computer system, wherein the kernel maintains kernel-level connection state information for the network connection, and wherein a socket layer maintains socket layer-level connection state information for the network connection;

   determining whether to offload the network connection from the [[first]] software network protocol stack to a ~~second~~ hardware network protocol stack implemented in a TCP Offload Engine (TOE)-capable network interface card operatively connected to the computer system;

   transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack using a network interface card driver when it is determined to offload the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack; and

   determining ~~whether~~ to accept ~~or refuse~~ the transfer of the network connection at the ~~second~~ hardware network protocol stack based ~~at least in part~~ on a [[state]] processing capability of the ~~second~~ hardware network protocol stack, wherein the network interface card maintains hardware-level connection state information for the network connection,[[;]] and ~~a nature of the network connection, wherein a determination to refuse the transfer further comprises refusing the transfer of the network connection based at least in part on the state of the second network protocol stack and the nature of the network connection, and wherein a determination to accept the transfer of the network connection at the second network protocol stack further comprises accepting~~

~~the transfer if the network connection exceeds the capability of the second network protocol network stack~~

wherein after the hardware network protocol stack accepts transfer of the network connection the software network protocol stack is configured to continually reference hardware-level connection state information and the hardware network protocol stack is configured to continually reference kernel-level connection state information and socket layer-level connection state information.

2. (Currently Amended) The method as recited in claim 1, ~~further comprising:~~ wherein transferring the network connection from the software network protocol stack to the hardware network protocol stack using the network interface card driver comprises:

obtaining, by the network interface card driver for the software network protocol stack, a hardware connection identifier maintained by the network interface card in association with hardware-level connection state information for the network connection, wherein the software network protocol stack is configured to use the hardware connection identifier to obtain hardware-level connection state information for the network connection; and

obtaining, by the network interface card driver for the hardware network protocol stack, a reference to socket layer-level connection state information and a reference to kernel-level connection state information, wherein the hardware network protocol stack is configured to use the references to create mappings from the hardware connection identifier to both socket layer-level connection state information and kernel-level connection state information, wherein the hardware network protocol stack is further configured to use the mappings to obtain kernel-level connection state information and socket layer-level connection state information for the network connection.

~~sharing state information associated with the network connection between the first network protocol stack and the second network protocol stack.~~

3. (Currently Amended) The method as recited in claim 1, wherein determining whether to offload the network connection is performed by [[an]] <u>the</u> operating system kernel of the computer system.

4. (Currently Amended) The method as recited in claim 3, wherein determining whether to offload the network connection is performed by [[a]] <u>the</u> socket layer of the operating system kernel.

5. (Currently Amended) The method as recited in claim 1, wherein determining whether to offload the network connection is performed by the [[first]] <u>software</u> network protocol stack.

6. (Canceled)

7. (Canceled)

8. (Canceled)

9. (Currently Amended) The method as recited in claim 1, wherein the ~~second~~ <u>hardware</u> network protocol stack is capable of determining whether to offload the network connection back to the [[first]] <u>software</u> network protocol stack.

10. (Currently Amended) The method as recited in claim 9, further comprising:

    receiving an indicator from the ~~second~~ <u>hardware</u> network protocol stack ~~or a driver associated with the second network protocol stack~~, the indicator indicating a request to transfer the network connection back to the [[first]] <u>software</u> network protocol stack.

11. (Currently Amended) The method as recited in claim 10, further comprising:

obtaining the hardware-level connection state information for the network connection from the ~~second~~ hardware network protocol stack ~~or~~ using the network interface card driver ~~associated with the second network protocol stack~~ and the hardware connection identifier for the network connection when the indicator is received; and

handling the network connection by the [[first]] software network protocol stack using the obtained hardware-level connection state information.

12. (Canceled)

13. (Currently Amended) The method as recited in claim 11, further comprising:

obtaining at least one of unsent and undelivered data by the [[first]] software network protocol stack from the ~~second~~ hardware network protocol stack ~~or a driver associated with the second network protocol stack~~, thereby enabling the [[first]] software network protocol stack to process the unsent or undelivered data.

14. (Canceled)

15. (Canceled)

16. (Canceled)

17. (Currently Amended) The method as recited in claim 9, further comprising:

handling the network connection by the [[first]] software network protocol stack [[when]] after the network connection is offloaded back to the [[first]] software network protocol stack from the ~~second~~ hardware network protocol stack.

18. (Canceled)

19. (Currently Amended) The method as recited in claim 1, further comprising:
    handling the network connection by the [[first]] software network protocol stack until it is determined to offload the network connection to the ~~second~~ hardware network protocol stack.

20. (Canceled)

21. (Canceled)

22. (Canceled)

23. (Canceled)

24. (Currently Amended) The method as recited in claim [[22]] 1, wherein the kernel-level connection state information comprises IP addresses and ports for a client and server of the network connection, and at least one of send and receive sequence numbers of one or more packets for the network connection.

25. (Currently Amended) The method as recited in claim 24, wherein the kernel-level connection state information further comprises:
    a round trip estimate.

26. (Currently Amended) The method as recited in claim 25, wherein the kernel-level connection state information further comprises:
    a congestion window and slow start information.

27. (Canceled)

28. (Currently Amended) The method as recited in claim 1, wherein upon transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack, the method further comprising:

    ~~at least one of~~ sending one or more inbound packets by the ~~second~~ hardware network protocol stack to the socket layer using the network interface card driver and receiving one or more outbound packets by the ~~second~~ hardware network protocol stack from the socket layer[[.]] using the network interface card driver, wherein the network interface card driver maintains a copy of each packet until the packet reaches its intended destination.

29. (Canceled)

30. (Currently Amended) An apparatus for processing a network connection in a computer system, comprising:

    means for establishing the network connection by a [[first]] software network protocol stack implemented in the kernel of an operating system associated with the computer system, wherein the kernel maintains kernel-level connection state information for the network connection, and wherein a socket layer maintains socket layer-level connection state information for the network connection;

    means for determining whether to offload the network connection from the [[first]] software network protocol stack to a ~~second~~ hardware network protocol stack stack implemented in a TOE-capable network interface card operatively connected to the computer system;

    means for transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack when it is determined to offload the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack, and

    means for determining ~~whether~~ to accept ~~or refuse~~ the transfer of the network connection at the ~~second~~ hardware network protocol stack based ~~at least in part~~ on a [[state]]

processing capability of the ~~second~~ hardware network protocol stack~~ and a nature of the network connection~~, wherein the network interface card maintains hardware-level connection state information for the network connection, and wherein after the hardware network protocol stack accepts transfer of the network connection the software network protocol stack is configured to continually reference hardware-level connection state information and the hardware network protocol stack is configured to continually reference kernel-level connection state information and socket layer-level connection state information~~ a determination to refuse the transfer further comprises refusing the transfer of the network connection based at least in part on the state of the second network protocol stack and the nature of the network connection, and wherein a determination to accept the transfer of the network connection at the second network protocol stack further comprises accepting the transfer if the network connection exceeds the capability of the second network protocol network stack~~.

31. (Currently Amended) A computer-readable medium storing thereon computer-readable instructions for processing a network connection in a computer system, comprising:

instructions for establishing the network connection by a [[first]] software network protocol stack implemented in the kernel of an operating system associated with the computer system, wherein the kernel maintains kernel-level connection state information for the network connection, and wherein a socket layer maintains socket layer-level connection state information for the network connection;

instructions for determining whether to offload the network connection from the [[first]] software network protocol stack to a ~~second~~ hardware network protocol stack implemented in a TOE-capable network interface card operatively connected to the computer system;

instructions for transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack using a network interface card driver when it is determined to offload the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack;[[,]] and

instructions for determining ~~whether~~ to accept ~~or refuse~~ the transfer of the network connection at the ~~second~~ hardware network protocol stack based ~~at least in part~~ on a [[state]] processing capability of the ~~second~~ hardware network protocol stack, wherein the network interface card maintains hardware-level connection state information for the network connection,[[;]] ~~and a nature of the network connection, wherein a determination to refuse the transfer further comprises refusing the transfer of the network connection based at least in part on the state of the second network protocol stack and the nature of the network connection, and wherein a determination to accept the transfer of the network connection at the second network protocol stack further comprises accepting the transfer if the network connection exceeds the capability of the second network protocol network stack~~ and wherein after the hardware network protocol stack accepts transfer of the network connection the software network protocol stack is configured to continually reference hardware-level connection state information and the hardware network protocol stack is configured to continually reference kernel-level connection state information and socket layer-level connection state information.

32. (Currently Amended) A network device comprising:

an operating system including a [[first]] software network protocol stack implemented in the kernel of an operating system associated with the computer system, wherein the kernel maintains kernel-level connection state information for a network connection, and wherein a socket layer maintains socket layer-level connection state information for the network connection;

a ~~second~~ hardware network protocol stack coupled to the [[first]] software network protocol stack, wherein the hardware network protocol stack is implemented in a TOE-capable network interface card operatively connected to the computer system, wherein the operating system being configured for determining whether to offload [[a]] the network connection to the ~~second~~ hardware network protocol stack and transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack using a network interface card driver when it determines that it will offload the network connection to the ~~second~~ hardware network protocol stack; and

a control component being configured for determining ~~whether~~ to accept ~~or refuse~~ the transfer of the network connection at the ~~second~~ hardware network protocol stack based ~~at least in part~~ on a [[state]] processing capability of the ~~second~~ hardware network protocol stack and wherein the network interface card maintains hardware-level connection state information for the network connection,[[;]] ~~and a nature of the network connection, wherein a determination to refuse the transfer further comprises refusing the transfer of the network connection based at least in part on the state of the second network protocol stack and the nature of the network connection, and wherein a determination to accept the transfer of the network connection at the second network protocol stack further comprises accepting the transfer if the network connection exceeds the capability of the second network protocol network stack~~ and wherein after the hardware network protocol stack accepts transfer of the network connection the software network protocol stack is configured to continually reference hardware-level connection state information and the hardware network protocol stack is configured to continually reference kernel-level connection state information and socket layer-level connection state information.

33. (Currently Amended) The network device as recited in claim 32, wherein the [[first]] software network protocol stack is a TCP/IP stack and the ~~second~~ hardware network protocol stack is a TCP/IP stack.

34. (Canceled)

35. (Canceled)

36. (Currently Amended) The network device as recited in claim 32, wherein the ~~second~~ hardware network protocol stack is capable of determining whether to offload the network connection back to the [[first]] software network protocol stack.

37. (Currently Amended) The network device as recited in claim 36, wherein the [[second]] hardware network protocol stack ~~or a driver associated with the second network protocol stack~~ sends an indicator when it requests to transfer the network connection back to the [[first]] software network protocol stack.

38. (Currently Amended) The network device as recited in claim [[32]]37, wherein the [[first]] software network protocol stack is adapted for obtaining hardware-level connection state information for the network connection from the ~~second~~ hardware network protocol stack [[or a]] using the network interface card driver and the hardware connection identifier for the network connection ~~associated with the second network protocol stack~~ when [[an]] the hardware connection indicator is received, thereby enabling the [[first]] software network protocol stack to handle the network connection using the obtained hardware-level connection state information.

39. (Canceled)

40. (Currently Amended) The network device as recited in claim 38, wherein the [[first]] software network protocol stack is further adapted for obtaining at least one of unsent and undelivered data from the ~~second~~ hardware network protocol stack, thereby enabling the [[first]] software network protocol stack to process the unsent or undelivered data.

41. (Canceled)

42. (Canceled)

43. (Canceled)

44. (Canceled)

45. (Currently Amended) The network device as recited in claim 36, wherein the [[first]] software network protocol stack is capable of handling the network connection when the network connection is offloaded back to the [[first]] software network protocol stack from the ~~second~~ hardware network protocol stack.

46. (Canceled)

47. (Currently Amended) The network device as recited in claim 32, wherein the [[first]] software network protocol stack handles the network connection until it is determined by the operating system to offload the network connection to the ~~second~~ hardware network protocol stack.

48. (Canceled)

49. (Canceled)

50. (Canceled)

51. (Canceled)

52. (Currently Amended) The network device as recited in claim 50, wherein the kernel-level connection state information comprises [[EP]] IP addresses and ports for a client and server of the connection, and at least one of send and receive sequence numbers of one or more packets for the connection.

53. (Currently Amended) The network device as recited in claim 52, wherein the kernel-level connection state information further comprises:
    a round trip estimate.

54. (Currently Amended) The network device as recited in claim 53, wherein the kernel-level connection state information further comprises:
    a congestion window and slow start information.

55. (Canceled)

56. (Currently Amended) The network device as recited in claim 32, wherein upon transferring the network connection from the [[first]] software network protocol stack to the ~~second~~ hardware network protocol stack, the ~~second~~ hardware network protocol stack is in communication with a socket layer of the [[first]] software network protocol stack, thereby enabling data to be sent by the ~~second~~ hardware network protocol stack to the socket layer and enabling data to be received by the ~~second~~ hardware network protocol stack from the socket layer.

57. (New) The method as recited in claim 1, wherein the network interface card driver is configured to maintain a copy of kernel-level connection state information and a copy of socket layer-level connection state information after the hardware network protocol stack accepts transfer of the network connection.

58. (New) The network device as recited in claim 32, wherein the network interface card driver is configured to maintain a copy of kernel-level connection state information and a copy of socket layer-level connection state information after the hardware network protocol stack accepts transfer of the network connection.